A Powerful Optimization Tool for Analog Integrated Circuits Design

Miloslav KUBAŘ [1,2], Jiří JAKOVENKO [2]

[1] ASICentrum spol. s r.o., a company of the Swatch Group, Novodvorská 994, 142 21 Prague, Czech Republic
[2] Dept. of Microelectronics, Czech Technical University, Technická 2, 166 27 Prague, Czech Republic

miloslav.kubar@asicentrum.cz, jakovenk@fel.cvut.cz

Abstract. *This paper presents a new optimization tool for analog circuit design. The proposed tool is based on the robust version of the differential evolution optimization method. Corners of technology, temperature, voltage and current supplies are taken into account during the optimization. That ensures robust resulting circuits. Those circuits usually do not need any schematic change and are ready for the layout. The newly developed tool is implemented directly to the Cadence design environment to achieve very short setup time of the optimization task. The design automation procedure was enhanced by optimization watchdog feature. It was created to control optimization progress and moreover to reduce the search space to produce better design in shorter time. The optimization algorithm presented in this paper was successfully tested on several design examples.*

Keywords

Analog circuit synthesis, optimization tool, differential evolution, search space, optimization watchdog.

1. Introduction

While the analog part of the integrated circuit covers usually 10 % of the chip area, its design takes about 90% of the time needed to design the whole circuit. An automated optimization of the analog circuits can save enormous part of that time. At present much effort is spent on development of an optimization tool to shorten time of the analog circuit design [1-5].

Our work is aimed to design the optimization tool for industry everyday design work. It requires a very short setup time of the design task, accurate ready-to-use results and a robust tool to be able to converge to the solution for a generic circuit and its specification (i.e. to not stuck during the optimization progress).

1.1 State-of-the-Art

The analog design automation approaches published so far are [6], [7]:

- Knowledge based - contains complete design plan describing how the circuit components must be sized to reach the solution of the design problem. There is no guarantee of finding the optimum solution.

Following three design automation approaches are optimization based. They use an optimization engine instead of a design plan to perform the design task. The optimization process is an iterative procedure. The design variables are updated at each iteration until they achieve an equilibrium point. The optimization algorithm searches through the search space for values of each circuit component. The performance evaluation tool verifies if the performance constraints are met:

- Equation based [3], [8] - use analytic design equations to evaluate the circuit performance. These equations can be derived manually or automatically by symbolic analysis tools. Then, the problem can be formulated as an optimization problem and normally solved using a numerical algorithm. The main drawback is that analytical models have to be used to derive the design equations for every new topology and, despite recent advances in symbolic circuit analysis, not all design characteristics can be easily captured by analytical equations. The approximations introduced in the analytical equations yield low accuracy design especially in complex circuit designs.
- Simulation based [9-11] – use simulations to evaluate the circuit performance. They extract circuit parameters to be optimized from the simulation results. This is pointed out as a very flexible solution when compared with other methodologies (equation-based, knowledge-based) once it accommodates to any type of circuit topology and yields superior accuracy (depends on simulator models). The same circuit can be optimized several times for different specifications as long as the fitness function is adapted. Therefore virtually all types of circuits can be optimized with this approach and with short setup time. The drawback of those methods is that they are computationally very expensive to evaluate performance of the optimized circuit by electrical simulations.
- Learning strategy based [1], [13] - the behavior of the circuit to be optimized is modeled by a learning

mechanism based on the distribution of variation. Thus they allow a quick evaluation of the performance for a specific set of design parameters. They require the set of training samples that must be evaluated at the beginning of the optimization. The amount of the training data will influence the accuracy of the performance predictions made by the learning machine. There will always be a trade-off between accuracy and efficiency in learning based optimization tools as in equation-based methods.

The equation based methods are not accurate enough to design automatically analog ready-to-use circuits. Learning based strategies can produce powerful circuits. But their setup time can be longer than a design without any optimization tool because of training samples creation. Simulation based tools produce the most accurate circuit and its setup time is the shortest. Therefore it is our choice for the proposed tool despite the fact that the computation time is longest in case of this approach.

Many works about the automated analog circuit design published recently are quite sophisticated and present powerful analog circuit synthesis ideas and improvements. On the other hand those works or the principles they present are not optimal solutions for the industry design optimization tool. Such a tool must satisfy the requirements of our approach:

- Very short setup time of the design task
- Accurate ready-to-use results.
- Robust tool to be able to converge to the solution for an arbitrary circuit.

Works [3], [9], [12] use equation based optimization method. They reach very fast optimization times usually by Matlab implementations. Nevertheless the equation based approaches cannot produce as accurate results as designers need even using sophisticated device models.

Automation algorithm presented by Pereira-Arroyo at al. [1] uses learning based method for much faster convergence progress. It uses accurate simulation to generate training samples. Neural networks optimization core quite quickly optimizes the analog circuits. That core is created in Matlab and does not use simulation based approach. Thus it does not generate accurate circuits.

Somani at al. [2] use powerful knowledge based initial setup of their automated design approach to enhance its convergence progress. It works well for specific design examples. On the other hand, this approach is not optimal for generic circuits and can result in a long setup time.

The method presented by Bo at al. [4] is a powerful optimization method based on a hybrid approach. It is a combination of Matlab equation based and Hspice simulation based approaches. That method achieved a very good optimization times even for complex circuits. Nevertheless it costs quite long setup time especially because of finding the penalty coefficients they use.

Barros at al. [13] use powerful learning mechanism that is combined with corner simulation based method. This combination leads to enhanced convergence of the tool. It leads to long setup time to create enough training samples that can pay off in complex design tasks. But the setup time is usually too long in typical design cases.

Algorithm presented by Fakhfakh at al. [5] uses Alienor-based method to significantly reduce the number of design variables. It leads to enhanced convergence of the optimization. Unfortunately that approach is not usable in simulation based optimization tool that needs all design variables to define for example lengths and widths of all transistors.

Optimization approach presented by Thakker at al. [10] is simulation based one. It uses accurate device models. It performs PVT (Process, Voltage, Temperature corners) simulations. To improve optimization time it runs the PVT simulation only for the final solution. The specification cannot be fulfilled in the worst case corner because of that approach.

The simulation based approach published by Prem Kumar at al. [11] is based on Particle Swarm Optimization. It is able to optimize to powerful solution but only for typical conditions and corners. It produces circuits with bad components matching (transistors in current mirrors do not have equal widths and lengths).

The powerful commercial tool [14] is used widely to partly automate design work. It is hard to design circuits purely automatically. It needs long setup time to create testbenches needed for the optimization and manual definition of the design task. The created design examples are also technology dependent. Moreover the optimization method is not under control as it is in our optimization method using the novel optimization watchdog feature.

State-of-the-art of the analog circuit design automation is well described in [6], [7]. Moreover open research points in this field are discussed in [6].

1.2 Proposed Approach

We propose a solution of one open point mentioned in [6] – to reduce search space of the optimization task (to reduce intervals of widths and lengths of transistors in our designs) by a novel optimization watchdog feature. It is described in Section 3. Its verification is listed in Section 4.1. The search space reduction leads to faster convergence time of the optimization progress and thus to the reduction of the computation time.

The optimization watchdog feature was implemented also to help the designer when the tool is not able to converge to proper solution. This can happen when the specification is too demanding or when the bounds of the design variables are set too tight. Our optimization tool is able to identify them and to advise the user of the proposed tool to set them better. It can also advise how to reduce the search

space for quicker convergence of the optimization progress.

We present a novel optimization tool implemented to Cadence design environment GUI (graphical user interface). It can be utilized in industry everyday work and can reduce the setup time of the automated design task. Integration of the optimization tool was implemented by SKILL language [15]. Ocean scripting language [15] was used to perform algorithms of the developed optimization tool. The proposed optimization tool uses the simulation approach with real device models (AMIS 0.35 μm technology was used but it can be switched to anyone) and full PVT analysis to produce robust ready-to-use circuits.

The presented optimization approach was successfully tested on the optimization of two OTA architectures and one voltage regulator architecture. They are planned (among others) in design of sigma delta modulators [16], [17]. Those circuits have been used as the test examples for the optimization tool. The presented tool is universal and can be used to optimize an arbitrary circuit (Section 4.4).

The article is organized to the five sections: Section 2 introduces the used optimization method and reasons for its usage. Section 3 describes the proposed optimization tool, its implementation to the Cadence design environment and the optimization watchdog feature with its advantages. Section 4 presents design examples used in our work and the result of their optimizations. An example of usage of the novel optimization watchdog feature to decrease optimization time is given in that section. It also contains comparison with other works presented so far. Current mirror transistor sizing approach we used is described there. It is more accurate than the approaches used in [18], [19], [20]. Finally, Section 5 gives the conclusions of this work.

2. Optimization Algorithm

Suitable optimization method is required to make a robust optimization tool. Gradient based methods that are frequently used in such process can be easily trapped in a local minimum. Much better performance could be achieved by evolutionary techniques. They are designed to converge to the global extreme because of their stochastic behavior [8]. These techniques are also well suited for multi-criterion optimization [9] which is the case of analog circuit optimization.

Differential evolution optimization algorithm was developed by Storn and Price [21]. It belongs to the group of evolutionary algorithms that work with functions of several real variables. The work was published in 1996 and in the same year differential evolution algorithm was presented at the First International Contest on Evolutionary Computation in Nagoya. Differential evolution began to be one of the best evolution algorithms for solving the real-valued test function [22], [23]. In the recent years differential evolution algorithm became the very good choice for analog circuit sizing [8], [24], [25] in terms of:

- Optimization convergence time.
- Optimization stability of non-convex, multi-modal and non-linear functions.
- Rapid convergence speed.
- Solving multi-variable real-valued functions.
- Operations of the differential evolution are simple and easy to program.

Differential evolution is similar to the overall structure of the genetic algorithm. The main difference is the mutation operation. This operation uses a perturbation of two members as the vector to add to the third member, which produces a new vector. The new vector is then mixed with the predefined parameters in accordance with certain rules to produce trial vectors. This operation is called crossover. If the trial vector fitness is less than the target vector fitness, the trial vector is placed instead of the target vector to the next generation. Those operations must be done for all members of the population in order to produce the same number of competitors in the next generation.

In past few years several improvements of the differential evolution were presented [22], [25] to improve its abilities. They enhance the algorithm usually despite of its other advantage, for example improving of convergence speed with danger of the convergence to the local extreme. This is not needed for the proposed robust optimization tool that must produce circuit as powerful as possible even with the cost of longer computation time (there is usually enough of machine time but not of analog designer time in the industry design). That is why we have chosen the simple but robust version of the differential evolution called DE/rand/1/bin [25] for our optimization approach.

Combinations of genetic algorithms and gradient-type methods were found to be more efficient than differential evolution itself. Guanglong at al. [26] presented a combined method of differential evolution and Gauss-Newton algorithm. Zhou at al. [27] published a combined method of specific genetic algorithm and Levenberg-Marquardt method. Both approaches combine advantages of genetic algorithm to converge to the global optimum and fast convergence speed of the gradient-type methods. Nevertheless we cannot use similar algorithms. The reason is that we cannot compute necessary derivations in simulation based optimization tool.

Differential evolution 4 basic steps can be described as follows:

Step 1: Initialization. The first step of the differential evolution creates arbitrary initial population of size n in the p-dimensional space as follows:

$$x_{ij}(0) = rnd_{ij}(0,1)(x_{ij}^{U} - x_{ij}^{L}) + x_{ij}^{L}, \quad i = 1,2,...,n; \quad j = 1,2,...,p \tag{1}$$

where x_{ij}^{U}, x_{ij}^{L} denotes to upper and lower bounds of the j^{th} design variable in the population respectively, rnd_{ij} (0,1)

represents a uniformly distributed random value within [0, 1].

Then the fitness of each vector/individual in the population is computed. More circuit parameters are optimized usually in the analog circuit optimization tasks. Therefore the fitness function is necessary for the circuit fitness determination. We use the fitness function presented in [10] that showed good optimization convergence speed and results:

$$F = \sqrt{\sum_{i=1}^{cp} X^2},$$
$$X = \frac{CP_{SPi} - CP_{SIMi}}{CP_{SPi}} \text{ for } CP_{SPi} > 0, \quad (2)$$
$$X = CP_{SIMi} \text{ for } CP_{SPi} = 0$$

where CP_{SP} represents the circuit parameter specification and CP_{SIM} denotes the simulation value of a circuit parameter. The sum is done for cp optimized circuit parameters. The circuit parameter which satisfies its specification does not contribute to that sum. Fitness function is equal to 0 if all simulated circuit parameters meet the specification. A circuit that is better than the other one has lower fitness function value than the worse circuit.

Step 2: Mutation operation. Differential evolution randomly selects two vectors of the population x_{p2}, x_{p3} ($p2 \neq p3$). It computes their difference and multiplies the difference by scaling factor F to mutate x_{p1} as follows:

$$h_{ij}(t+1) = x_{p2j} + SF(x_{p2j}(t) - x_{p3j}(t)) \quad (3)$$

where $x_{p2j}(t)$ - $x_{p3j}(t)$ is the differential vector and SF (real number from interval [0, 2] [21]) is the scaling factor.

Step 3: Crossover operation. In order to increase the diversity of population, crossover operation is introduced to generate a trial vector:

$$v_{ij}(t+1) = h_{ij}(t+1) \quad \text{for } (rnd1_{ij} \le CR),$$
$$v_{ij}(t+1) = x_{ij}(t) \quad \text{for } (rnd1_{ij} > CR) \quad (4)$$

where rnd_{ij} is a uniform random number within [0,1]. CR denotes the crossover constant also within [0,1].

Step 4: Selection operation. The trial vector $v_i(t+1)$ is compared with the target vector $x_i(t)$ using evaluating function value:

$$x_i(t+1) = v_i(t+1) \text{ for F}(v_{i1(t+1)},...,v_{in}(t+1) < F(x_{i1(t+1)},...,x_{in}(t+1))$$
$$x_i(t+1) = x_i(t) \text{ for F}(v_{i1(t+1)},...,v_{in}(t+1) \ge F(x_{i1(t+1)},...,x_{in}(t+1)) \quad (5)$$

The process from step 2 to step 4 is then repeated through until the function meets the specified value. The flow diagram of the differential evolution is shown in Fig. 1.

Consequences of the evolutionary approaches of the analog circuit optimization task are high demands on disk space and possible overflow of the computer memory. The reason is lot of results of the circuit simulations. It can happen if the optimization converges to the solution slowly thus lot of simulation results are stored in the computer memory and disk. Those problems are solved in the developed optimization tool.

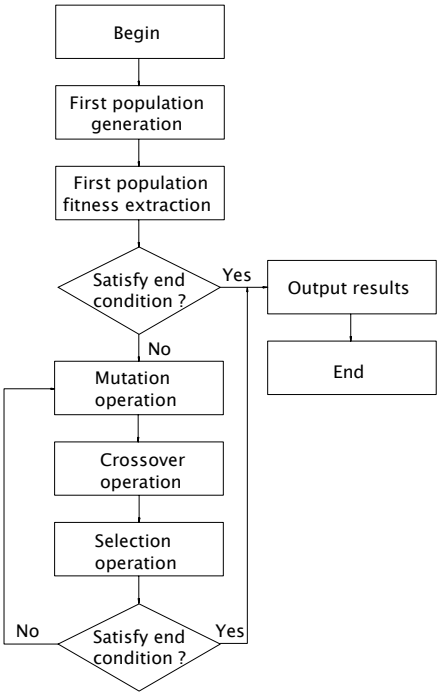

Fig. 1. Differential evolution flow diagram.

3. Optimization Tool

The optimization core together with implementation of the optimization method described in Section 2 was designed using Ocean scripting language. Designed scripts enable:

- Implementation of the optimization algorithm.
- Spectre circuit simulations required by the optimization method.
- Post-processing of the simulations output to extract circuit optimized parameters.

The core interface is implemented to the Cadence GUI by the Skill language to be easily improved or expanded in the future. Cadence was chosen because it is frequently used analog design environment. That kind of implementation makes the setup time of our optimization very short (few tens of seconds). Short setup time is one of the main criterions of the designed optimization tool. The flow diagram of the optimization tool is shown in Fig. 2.

Analog integrated circuits are designed in Cadence usually in the following way. The circuit schematic is created using the CAD tool. Then the testbenches of the circuit needed to simulate specified circuit parameters are created. The testbenches are used to tune the circuit until the specification is met. All above mentioned tasks are not required in the circuit design using our optimization tool.

The GUI is used only to select the circuit type to be optimized, enter the circuit specification and run the optimization procedure. The SKILL script was created to insert new "optimize" toolbar to the Cadence main window. The circuit which has to be optimized can be selected together with the definition of circuit parameters in that toolbar. The specification of the circuit is send to the optimization core as a text file in the format of Ocean scripting language.

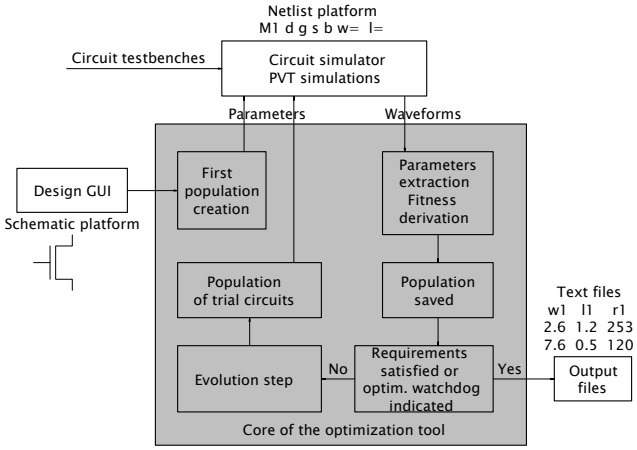

Fig. 2. Algorithm of the optimization tool.

The optimization core uses pre-created netlist of the optimized circuit test-benches to run Spectre simulation. The design variables (as transistors widths and lengths) are sent to circuit simulator by text file in the Spectre simulator format. Output of the optimization tool is text files containing the details of all the circuits in all created populations.

The main optimization tool core script is shown in Fig. 3. It calls several second-order scripts that are not included in this article because they cover together more than 1000 lines.

First of all the initialization of the optimization tool is made (scripts *declaration.ocn* and *skill_out.ocn*). Design variables bounds, parameters of the differential evolution (n, SF, CR) are set. Moreover design specification is loaded from *skill_out.ocn* script. That script is created by the Skill script of the optimization tool interface. The last step of the initialization phase is the opening of the output files (design variables values and circuit parameters values).

Then the first population is created (*first_params.ocn* script). PVT Spectre simulations of the circuits in the first population are run. The worst case optimized circuit parameters are extracted from the simulations results (*first_sim.ocn* script). The details about the first population are stored in the output files (*output_data.ocn* script). The fitness of the circuits in the first population is computed and checked if the specified circuit was found. The optimization process continues until the solution is found.

Only the worst case corner simulations are run for each circuit parameter to speed up the optimization. It was needed to run the optimization for all corners to determine the worst case for each circuit parameter. It was done during the design examples creating (only one population of one individual was enough to specify the worst case).

```
load("/2s_ota/skill_out.ocn")
load("/2s_ota/declaration.ocn")

of=outfile("/2s_ota/results/outputfile.scs"))
xff=outfile("/2s_ota/results/xfactor_file.scs"))

for(i 1 n
   load("/2s_ota/first_params.ocn"))
   load("/2s_ota/first_sim.ocn")
   load("/2s_ota/output_data.ocn")
   done = if(((F[i]=0.0)||(done==1)) 1 0)
)

z = z + 1

while((done<1)

   for(i 1 n
      load("/2s_ota/random.ocn")

      load("/2s_ota/param_evo.ocn"))
      load("/2s_ota/evo_sim.ocn"))

      delta = F[i] - Ftmp
      stuck = if(((delta>=wdd)||(stuck==0)) 0 1)

      load("/2s_ota/evo_new_population.ocn"))
      done = if(((F[i]=0.0)||(done==1)) 1 0)
      load("/2s_ota/output_data.ocn"))
   )

   z = z + 1
   wdc = if((stuck==1) (wdc+1) 0)
   stuck = 1
   done = if((wdc>=wdp) 1 done)

)

load("/2s_ota/final_output.ocn"))
close(of)
close(xff)
```

Fig. 3. Ocean script – the optimization tool core.

The procedure of the differential evolution (mutation, crossover and selection) is done in the second *"for"* loop. Three random numbers are computed (*random.ocn* script), design variables of the trial circuits are generated (*param_evo.ocn* script) and trial circuits are simulated. Their fitness is computed and new population is created (*evo_new_population.ocn* script). Again the details of the new population are stored to the output files. The *"while"* loop is run again until the final condition is satisfied.

The first final condition is the occurrence of a circuit with fitness function equal to zero – goal of the optimization. Another final requirement is the occurrence of a predefined number of populations (parameter WD_P) in a row without significant progress of the optimization. It is defined by specific difference between fitness of the trial and target circuit – parameter WD_D. This indicates that the optimization is not able to get much better circuit in a reasonable time. This is the baseline of the novel feature – optimization watchdog – that helps to optimize a better circuit in a shorter time. The watchdog is implemented as follows:

$$WD_C = 0 \quad \text{if } \exists i = 1,...,n \;\; F_i(t+1) \le F_i(t) - WD_D$$
$$WD_C = WD_C + 1 \quad \text{otherwise} \tag{6}$$

where WD_C, WD_D and WD_P are special watchdog variables, n is the number of individuals in one population. The optimization ends without satisfying the specification if WD_C is equal to WD_P.

WD_D and WD_P parameters are set to their default values by the optimization tool. On the other hand, the setting can be changed by the tool user. WD_C variable is evaluated during the optimization process by the tool as shown in Fig. 3.

The first reason for optimization watchdog implementation is the natural feature of the differential evolution. The optimization process can diverge (Section 2) if the specification of the circuit is set beyond its limits. It can also diverge if the bounds of the design variables are set too high or too low. If several populations without significant individual improvement occur the optimization is stopped consecutively.

The main reason for the optimization watchdog implementation is the convergence time improving by a reducing of the search space. The output file generated by *final_output.ocn* script contains information about design variables bound settings and indication how the bounds should be improved. The ranges of some design variables are narrowed down to reduce the search space to improve convergence speed of the optimization.

The simplified Ocean script that recommends how to modify the design variables bounds is shown in Fig. 4. First, individuals in the final population are sorted from the best one to the worst one. Then m variable is counted. It denotes to the number of the best circuits taken into account in the bounds modification. We chose cp to be 50 % of n (rounded down). It is a tradeoff between optimization optimality in terms of convergence to the local minimum danger and acceleration. The last step of the script creates new bounds of the design variables. Fig. 4 shows an example with only one design variable $w1$.

The optimization watchdog first generates WD_P low (from 1 to 2 in dependence on the circuit complexity – 1 for circuit with about 5 design variables and 2 for circuit with 15 design variables) and WD_D high (from 0.05 to 0.1 – lower value for higher number of design variables) for short optimization which quickly scans the circuit, its specification and setting of the design variables bounds.

Then the design variables bounds are improved in accordance to the output of the first short optimization and the circuit requirements (using the script shown in Fig. 4). WD_P is set higher (from 3 to 5) and WD_D lower (from 0.01 to 0.03) for precise optimization that is able to achieve powerful circuits.

The second task performed by the watchdog is to identify the design variables bounds set inappropriately. It recommends what bounds should be extended to achieve enhanced circuit performance.

The specification can be found to be beyond the limit of the circuit after the first short optimization. At that point the specification can be changed for example by some tradeoff between consumption and slew-rate of the circuit.

```
m = n - 1
Ftmp=F[1]
for(i 1 m
        for(j 1 m
                while((F[j+1]<F[j])

                Ftmp=F[j]
                F[j]=F[j+1]
                F[j+1]=Ftmp

                w1tmp=w1[j]
                w1[j]=w1[j+1]
                w1[j+1]=w1tmp)))

if((mod(n 2)>0) m=((n-1)/2) m=(n/2))

w1min_new = w1[1]
w1max_new = w1[1]

for(i 2 m

if((w1[i]<w1min_new) w1min_new=w1[i])
if((w1[i]>w1max_new) w1max_new=w1[i]))
```

Fig. 4. Ocean script – design variables bounds modification.

4. Circuit Optimization and Results

The presented optimization tool was tested on optimization of two-stage Miller OTA (Section 4.1). This circuit was also used to compare the performance with other works since it is the most frequently used circuit in the works published recently.

Comparison of our work with other optimization approaches presented so far was a difficult task as all details needed to compare the results (as bounds of the design variables, supply voltage, bias current, capacitive load of the circuit and optimization time) were not presented (works [8], [11], [18], [19], [20]). Comparison details are listed in Section 4.5.

Other design examples - folded cascode OTA (Section 4.2) and voltage regulator (Section 4.3) – were created to be used in sigma-delta modulator [16], [17].

Section 4.4 reports on how the optimization tool can be extended for other analog circuits.

All circuits were designed using AMIS 0.35 μm CMOS technology device models. We obtained the models from the foundry as a part of the design kit. The models are referenced in the core script. The technology can be easily changed if required by changing of the references in the script. All optimizations were run for temperature range -10 °C to 50 °C, supply (input) voltage range 1.8 V to 2.0 V and bias current variations of 30%. The optimization

were run on 2.5 GHz two core Intel processor with 4 GB RAM. Most of the optimization time was consumed by Spectre circuit simulations. 7.0.1.091 Spectre simulator version was used.

Population size n was chosen to 15 individuals for all optimized circuits. It was proven to be large enough for the optimization convergence and small enough for the optimization speed. Scaling factor SF was set to 0.8 and crossover constant CR to 0.7. Those values were found as a good compromise between the optimization convergence speed and the possibility to obtain powerful circuits.

4.1 Two-stage Miller OTA

Circuit schematic is shown on Fig. 5 with design variables underlined.

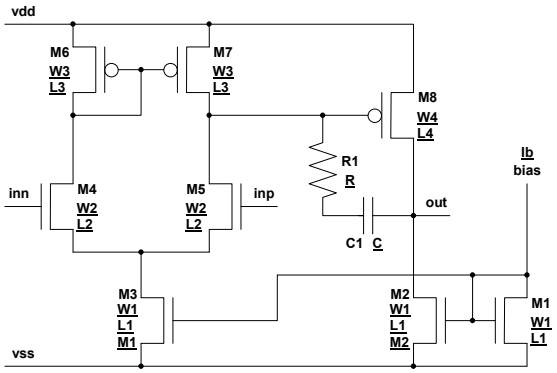

Fig. 5. Two-stage Miller OTA.

Width and length of transistors M1, M2 and M3 (parameters $W1$ and $L1$) are not strictly optimized but derived from bias current (to have appropriate area because of good matching and specific operation point because of good mirroring ratio). The number of design variables is 11. Load of the OTA is 10 pF.

Accurate approach to size the current mirror transistors was used. It is based on using the same width and length of all current mirror transistors (transistors M1, M2 and M3) and multiplying transistors in parallel to increase or decrease bias current in each part of the circuit (by design variables $M1$ and $M2$). It is much better approach to transistor matching than to size widths and lengths of the current mirror transistor independently as presented in [11], [18], [19], [20].

First, the verification of the optimization acceleration by using of the optimization watchdog is listed. Demanding specification of the circuit was set and the optimization watchdog parameters were set 1 for WD_P and 0.1 for WD_D for the first rough optimization. The bounds of the design variables were updated in accordance to the result of the first optimization (to cover all values from the best half of the individuals in the last population). The parameters WD_P and WD_D were set to 3 and 0.05 respectively for the second optimization with the reduced search space. Parameter WD_P was set to 5 and parameter WD_D to 0.01 for the simulation without using of the watchdog.

Tab. 1. contains information about the circuit specification, optimization results and optimization results without using of the watchdog. The optimization time of the first watchdog optimization was 120 minutes (solution found in the 10th population). The second watchdog optimization lasted 276 minutes (solution found in the 23rd population). Thus the complete optimization time was 396 minutes with using of the watchdog. The optimization time was 960 minutes (solution found in the 80th population) without using of the watchdog. Thus the optimization time was reduced more than two times. Design variables with their bounds before/after the reduction of the search space and optimization results values are in Tab. 2.

Param.	Gain	PM	GBW	SR	Cons.
Spec.	90 dB	60 °	2.0 MHz	2.0 V/µs	20 µA
Result - WD	90 dB	64 °	2.8 MHz	2.3 V/µs	16 µA
Result - ~~WD~~	94 dB	60 °	2.4 MHz	2.2 V/µs	13 µA

Tab. 1. Circuit parameters - two-stage Miller OTA.

Variable	Lower bound	Upper bound	Result - WD	Result - ~~WD~~
W1 (µm)	0.4	50	2.6	1.6
L1 (µm)	0.55	50	1.6	2.7
W2 (µm)	0.5/10	50	37.2	50.0
L2 (µm)	0.5	20	1.7	5.5
W3 (µm)	0.5/8	50	41.0	18.1
L3 (µm)	0.5/13	50	49.4	50.0
W4 (µm)	0.5/9	50	38.4	46.5
L4 (µm)	0.5	10/7	0.7	0.5
R (k Ω)	0.1	10	0.1	0.1
C (pF)	0.1	10/5	0.58	0.38
M1 (-)	1	10	1	2
M2 (-)	1	10	2	6
Ib (µA)	0.1	30/9	4.0	1.2

Tab. 2. Design variables – two-stage Miller OTA.

Large differences between the two optimized circuits are apparent. The reason of these differences can be seen in existing of more comparable solutions (see Section 4.5 for more details).

4.2 Folded Cascode OTA

Circuit schematic is shown in Fig. 6 with design variables underlined. Width and length of current mirror transistors M1, M2, M3, M4, M5, M6, M7 and M8 (parameters $W1$, $L1$, $W2$ and $L2$) are derived in the same way as in the

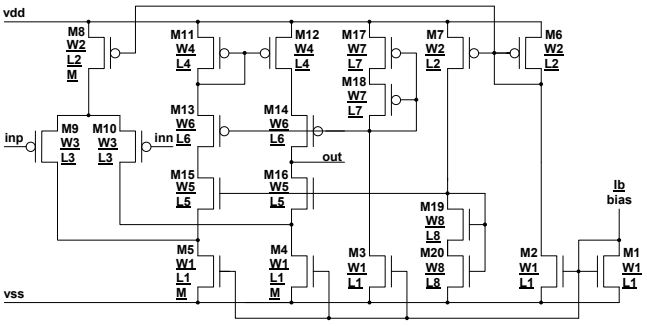

Fig. 6. Folded cascode OTA.

design example in Section 4.1. The number of design variables is 14. WD_P was set to 5 and WD_D to 0.03 because of

quite high number of design variables. Load of the OTA is 10 pF.

Specification and optimization results of the circuit parameters of the OTA are listed in Tab. 3.

Param.	Gain	PM	GBW	SR	Cons.
Spec.	50 dB	60 °	0.5 MHz	0.2 V/µs	50 µA
Result	68.0 dB	86.5 °	0.58 MHz	0.21 V/µs	35.8 µA

Tab. 3. Circuit parameters - folded cascade OTA.

Variable	Lower bound	Upper bound	Result
W1 (µm)	0.4	50	2.9
L1 (µm)	0.55	50	1.1
W2 (µm)	0.4	50	7.4
L2 (µm)	0.55	50	0.55
W3 (µm)	1	50	28.4
L3 (µm)	0.5	5	1.1
W4 (µm)	1	20	19.7
L4 (µm)	1	20	6.9
W5 (µm)	0.5	50	7.4
L5 (µm)	0.5	10	17.6
W6 (µm)	0.5	50	6.5
L6 (µm)	0.5	10	3.4
W7 (µm)	0.5	10	1.3
L7 (µm)	0.5	50	22.8
W8 (µm)	0.5	10	2.9
L8 (µm)	0.5	50	10.2
M (-)	1	9	1
Ib (µA)	0.1	10	17.0

Tab. 4. Design variables – folded cascade OTA.

Output voltage swing was also checked by optimization tool since it is very problematic parameter for this circuit. It was not optimized but only checked (by the slew-rate simulation) that it is higher than V_{VDD}-400 mV. If the swing is lower the circuit is considered as not satisfying the specification (fitness function set to 10).

Design variables with their bounds and optimization results values are in Tab. 4. Optimization took 123 minutes (solution found in the 5th population). It can be seen that the specification is less challenging than in two-stage OTA case. It was quite difficult to optimize this circuit. The reason for these difficulties is higher number of design variables than in two-stage OTA case.

4.3 Voltage Regulator

Circuit schematic is shown in Fig. 7 with design variables underlined.

Width and length of current mirror transistors M1 and M2 (parameters *W*1 and *L*1) are derived in the same way as in the design example in Section 4.1. Resistance of devices R1 and R2 (parameters *R*1 and *R*2) are set by reference voltage, typical output voltage and current consumption of the resistor divider (set to 20 µA). The number of design variables is 8. Load current range of the regulator is between 20 µA and 2 mA. Reference voltage range is between 1.23 V and 1.24 V. Capacitive load of the regulator is 500 pF with 5Ω series resistor. WD_P was set to 3 and WD_D to 0.05 in this case.

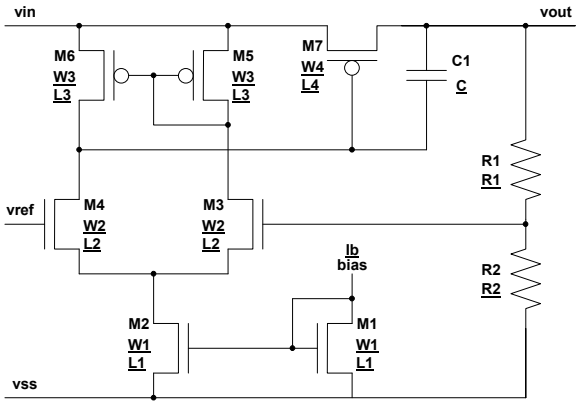

Fig. 7. Voltage regulator.

Circuit specification and optimization results of the circuit parameters of the optimized voltage regulator are shown in Tab. 5.

Param.	Vreg-typ	Vreg-var	Line reg.	Load reg.
Spec.	1.5 V	50 mV	30 mV	30 mV
Result	1.5 V	7 mV	0.4 mV	1.2 mV
Param.	PM	BW	Temp. drift	PSRR (50 Hz)
Spec.	60 °	150 kHz	0.5 mV/°C	40 dB
Result	91.3 °	171 kHz	3 µV/°C	51.6 dB

Tab. 5. Circuit parameters – voltage regulator.

Design variables with their bounds and optimization results values are in Tab. 6.

Lower and upper bounds of variables *R*1 and *R*2 are not needed since values of those variables depend just on the circuit specification (input voltage, output voltage and current consumption of the resistor divider).

Optimization time was 48 minutes (solution found in the 5th population). Optimization had very good progress since the number of optimization parameters is low even despite of high number of optimized circuit parameters.

Variable	Lower bound	Upper bound	Result
W1 (µm)	0.4	50	0.4
L1 (µm)	0.55	50	8.2
W2 (µm)	1	20	12.2
L2 (µm)	1	20	10.2
W3 (µm)	1	20	18.7
L3 (µm)	1	20	10.2
W4 (µm)	250	500	434.8
L4 (µm)	0.5	1	0.5
C (pF)	0.1	10	0.85
R1 (kΩ)	Depends just on circuit specifications		13.25
R2 (kΩ)			61.75
Ib (µA)	0.1	10	0.1

Tab. 6. Design variables – voltage regulator.

4.4 Generic Circuit

This section summarizes the actions needed to extend the proposed optimization tool to be able to optimize other kinds of analog circuits. The following items must be done:

- Creations of the circuit test-benches netlists.
- Definition of the circuit parameter extracts.

- Definition of the design variables.
- Worst cases of the circuit parameters (optional to speed up the circuit optimization).
- New menu item in the optimization tool interface.

4.5 Comparison with Other Works

The proposed optimization tool was compared with works [8], [11] using Miller two-stage OTA design example. These works present results with less accurate current mirrors matching approach that our one described in Section 4.1.

First, the comparison with work [8] is given. That work consists of comparison of various optimization algorithms. We compared our optimization tool with the result of the most similar optimization algorithm given in [8]. The comparison is on the other hand not easy since the load capacitance of the sized circuit in [8] is not mentioned. We chose 1 pF for our optimization. WD_D was set to 0.05 and WD_P to 3. Supply voltage of the circuit is 2 V. The results of the comparison are in Tab. 7 and in Tab. 8.

Tab. 7 and 8 contain two different optimization results found for same conditions. Stochastic behavior of the optimization algorithm is apparent from difference of these solutions. It also explains the reason why our solutions are different from the one presented in [8]. Another reason is that the specification is not a global optimum of the optimization task. Thus more different solutions exist for the given specification.

Work [8] uses $SF = 0.5$, $CR = 0.8$ and $n = 30$. Lengths of the transistors are not swept and are set to 1 μm. Resistance $R1$ is not used (thus it is set to 0 Ω in our case). Transistors M1 and M2 has the same size, transistor M3 has higher W (207 μm) and the same L that can lead to inaccurate mirroring. We solved it in our case by setting variable $M1$ to 1 and to sweep variable $M2$.

Param.	Gain	PM	GBW	SR	Cons.
Spec.	70 dB	60 °	1.0 MHz	1.0 V/μs	75 μA
Our result 1	86 dB	68 °	2.5 MHz	1.0 V/μs	42 μA
Our result 2	86 dB	66 °	3.3 MHz	1.7 V/μs	28 μA
Result [8]	92 dB	63 °	1.7 MHz	1.0 V/μs	70 μA

Tab. 7. Circuit parameters – comparison with [8].

Variable	Lower bound	Upper bound	Our result	Result [8]
W1 (μm)	5	20	2.8/2.4	17.0
W2 (μm)	50	150	71.4/65.6	84.5
W3 (μm)	5	20	12.3/8.2	12.6
W4 (μm)	50	250	158.6/186.2	141.5
C (pF)	0.1	10/5	3.7/2.0	2.6
M2 (-)	1	10	4/4	N/A
Ib (μA)	1	20	6.15/5.27	2.03

Tab. 8. Design variables – comparison with [8].

The differences between their and our case are that they optimize CMRR (Common Mode Rejection Ratio) to 90 dB and run just typical simulations. We did not optimize CMRR but use PVT simulation to obtain our result (the results in Tab. 7 are typical results in comparison with our worst case results). Their optimization time was 32 minutes and our one 51 and 63 minutes (solution found in the 5th and 6th population) that is a very good result for optimization using PVT simulations.

The second comparison was done with work [11] that is really recent work. It describes a powerful simulation based optimization approach based on Particle Swarm Optimization algorithm. The comparison was again not easy since the results presented in that article lack the information about design variables bounds, value of the bias current and optimization time of their design example. It can be compared, at least, if our optimization tool can size as a powerful circuit as in [11] and if it can be optimized in a reasonable time. Since no specification was mentioned in [11] results of its optimization are used as a specification.

The results of the comparison with [11] are in Tab. 9. Design variables with their bounds before/after the reduction of the search space and optimization results values are in Tab. 10. The tables contain two optimization results where two runs of the second watchdog optimization were performed. Another explanation of the result difference is our different current mirror sizing algorithm mentioned in Section 4.1.

Kumar at al. [11] presents the powerful circuit thus we optimized only for typical condition (we were not able to get the same circuit parameters with PVT analysis). Simulation time achieved by developed optimization tool and watchdog feature was 186 and 192 minutes. First watchdog optimization ended by the 11th population. Solutions were found in the 82nd and 85th populations of the second watchdog optimizations. It is a very good result if we take into account that bounds of the design variables were not available and were set very liberally.

Param.	Gain	PM	GBW	SR	Cons.
Spec.	81 dB	60 °	100 MHz	288 V/μs	1819 μA
Our result 1	82 dB	63 °	123 MHz	307 V/μs	1817 μA
Our result 2	82 dB	99 °	122 MHz	299 V/μs	1814 μA
Result [11]	81 dB	60 °	N/A	288 V/μs	1819 μA

Tab. 9. Circuit parameters – comparison with [11].

Variable	Low. bound	Up. bound	Our result	Result [11]
W1 (μm) M1	0.4	100	62.5/36.1	310.0
L1 (μm) M1	0.55	50	0.55/0.55	4.6
W1 (μm) M2	0.4	100	62.5/36.1	432.0
L1 (μm) M2	0.55	50	0.55/0.55	4.1
W1 (μm) M3	0.4	100	62.5/36.1	309.0
L1 (μm) M3	0.55	50	0.55/0.55	2.0
W2 (μm)	5/10	500	87.1/50.1	199.0
L2 (μm)	0.35/1	20/15	6.7/2.9	0.6
W3 (μm)	5/50	500/400	400.0/400.0	245.0
L3 (μm)	0.35/5	20	8.4/13.9	1.3
W4 (μm)	5/20	500	331.8/343.4	360.0
L4 (μm)	0.35	20/11	0.4/0.4	1.4
C (pF)	0.5/5	50	5.0/5.0	3.2
R (kΩ)	0.5	50/30	18.16/15.08	0.545
M1 (-)	1	10	2/3	N/A
M2 (-)	1/2	10	4/7	N/A
Ib (μA)	5/50	500/300	272.8/157.7	N/A

Tab. 10. Design variables – comparison with [11].

WD_D was set to 0.03/0.01 and WD_P to 1/5 for the first/second optimization respectively. Supply voltage was set to 3.3 V and load capacitance to 1 pF.

5. Conclusions

A novel approach of analog circuit optimization was presented in this paper. The optimization tool is implemented to the Cadence design environment to have a very short setup time. Another reason of this implementation is easy application of the tool. Simple differential evolution method is used as the optimization algorithm to be robust and thus to be able to converge to the solution for every design example.

Novel optimization watchdog feature was implemented. It is able to automatically change bounds of the search space. The main purpose is to reduce the search space and thus to shorten convergence time or to get better results.

PVT corner simulations are used during the circuit optimization. It causes higher accuracy of the results thus optimized circuits are usually ready for layout.

It was proven that this tool is able to generate optimized circuits without any need of creating schematics and test benches by optimization of the two-stage Miller OTA, folded cascade OTA and voltage regulator. Moreover the proposed tool was successfully compared with recently published works.

Next step of our work will be extending the tool to enable optimization of more types of circuits (for example voltage references or current sources).

Acknowledgements

This work has been supported by the Grant Agency of the Czech Republic project No. 102/09/160, CTU SGS grant No. SGS11/156/OHK3/3T/13 and the Ministry of the Interior grant No. VG20102015015.

References

[1] JARAFI, A., SADRI, S., ZEKRI, M. Design optimization of analog integrated circuits by using artificial neural networks. In *International Conference of Soft Computing and Pattern Recognition*, 2010, p. 385-388.

[2] SOMANI, A., CHAKRABARTI, P. P., PATRA, A. An evolutionary algorithm-based approach to automated design of analog and RF circuits using adaptive normalized cost functions. *IEEE Transactions on Evolutionary Computation*, 2007, vol. 11, no. 3, p. 336-353.

[3] BENNOUR, A., SALLEM, A., KOTTI, M., GADDOUR, E., FAKHFAKH, M., LOULOU, M. Application of the PSO technique to the optimization of CMOS operational transconductance amplifiers, *5th Internat. Conf. on Design and Technology of Integrated Systems in Nanoscale Era*, 2009, p. 1-5.

[4] BO, L., YAN, W., ZHIPING, Y., LEIBO, L., MIAO, L., ZHENG, W., JING, L., FRANCISCO, V. F. Analog circuit optimization system based on hybrid evolutionary algorithms. *Integration,* 2009, vol. 42, no. 2, p. 137-148.

[5] FAKHFAKH, M. A novel Alienor-based heuristic for the optimal design of analog circuits. *Microelectronics Journal*, 2009, vol. 40, no. 1, p. 141-148.

[6] TLELO-CUAUTLE, E., GUERRA-GOMEZ, I., DUARTE-VILLASENOR, M. A., DE LA FRAGA, L. G., FLORES-BECERRA, G., REYES-SALGADO, G., REYES-GARCIA, C. A., RODRIGUEZ-GOMEZ, G. Applications of evolutionary algorithms in the design automation of analog integrated circuits. *Journal of Applied Sciences*, 2010, vol. 10, p. 1859-1872.

[7] BARROS, M. F. M., GUILHERME, J. M. C., HORTA N. C. G. State-of-the-art on analog design automation. *Studies in Computational Intelligence*, 2010, vol. 294, p. 19-47.

[8] SABAT, S. L., KUMAR, K. S., UDGATA, S. K. Differential evolution and swarm intelligence techniques for analog circuit synthesis. In *World Congress on Nature & Biologically Inspired Computing*, 2009, p. 469-474.

[9] MARŠÍK, J., ŠUBRT, O., MARTINEK, P. Developing automated design procedure for operational amplifier blocks. In *International Conference on Signals and Electronic Systems*, 2008, p. 269-272.

[10] THAKKER, R. A., BAGHINI, M. S., PATIL, M. B. Low-power low-voltage analog circuit design using hierarchical particle swarm optimization. In *22nd International Conference on VLSI Design*, 2009, p. 427-432.

[11] PREM KUMAR, P., DURAISWAMY, K. An optimized device sizing of analog circuits using particle swarm optimization. *Journal of Computer Science*, 2012, vol. 8, no. 6, p. 930-935.

[12] DIMOV, B., BOOS, V., REICH, T., LANG, C., HENNIG, E., SOMMER, R. A novel technique for CAD-optimization of analog circuits with bipolar transistors. *Advances in Radio Science*, 2009, vol. 7, p. 219-223.

[13] BARROS, M., GUILHERME, J., HORTA, N., Analog circuits optimization based on evolutionary computation techniques. *Integration*, 2009, vol. 43, no. 1, p. 136-155.

[14] Cadence Design Systems, Inc., Virtuoso NeoCircuit. Cited 2012-11-14. Available at http://www.cadence.com/rl/Resources/-datasheets/VirNeoCircuit_ds.pdf.

[15] Ocean reference. [Online] Cited 2013-02-21. Available at: https://secure.engr.oregonstate.edu/wiki/ams/files//Cadence.WritingCadenceOCEANScripts/oceanref.pdf.

[16] PAVLÍK, M., HÁZE, J., VRBA, R. Switched current 12-bit capacity to digital delta-sigma modulator. In *International Conference on Advances in Electronics and Micro-electronics*, 2008, p. 192 - 196.

[17] FUJCIK, L., MICHAELI, L., HÁZE, J., VRBA, R. Sensor signal digitization utilizing a band-pass sigma-delta modulator. *IEICE Transactions on Electronics*, 2009, vol. E92.C, no. 6, p. 860 - 863.

[18] JIANHAI, Y., ZHIGANG, M. Automated design method for parameters optimization of CMOS analog circuits based on adaptive genetic algorithm. In *7th International Conference on ASIC*, 2007, p. 1217-1220.

[19] MEDURI, P. K., DHALI, S. K. A methodology for automatic transistor-level sizing of CMOS opamps. In *Annual Conference on VLSI Design*, 2011, p. 100-105.

[20] MEDURI, P. K., DHALI, S. K. A framework for automatic CMOS opamp sizing. In *53rd IEEE International Midwest Symposium on Circuits and Systems*, 2010, p. 608-611.

[21] STORN, R., PRICE, K. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 1997, vol. 11, no. 4, p. 341–359.

[22] GAO-YANG, L., MING-GUANG, L. The summary of differential evolution algorithm and its improvements. In *3rd International Conference on Advanced Computer Theory and Engineering*, 2010, vol. 3, p. V3-153 – V3-156.

[23] *Differential Evolution.* [Online] Cited 2012-11-14. Available at: http://www.icsi.berkeley.edu/~storn/code.html.

[24] MARTINEK, P., TICHA, D. The improved DE algorithm for filter design. In *Radioelektronika*, 2008, p. 1-4.

[25] YOUYUN, A., HONGQIN, C. Experimental study on differential evolution strategies. In *Global Congress on Intelligent Systems*, 2009, vol. 2, p. 19-24.

[26] GUANGLONG, X., JISHUANG, X. A hybrid method for electromagnetic propagated resistivity logging data inversion. *IEEE Transactions on Geoscience and Remote Sensing*, 2007, vol. 45, no. 3, p. 649-655.

[27] ZHOU, Q., YAO, W., WU, W., LI, X., ZHU, Z., GILDENBLAT, G. Parameter extraction for the PSP MOSFET model by the combination of genetic and Levenberg-Marquardt algorithms. In *IEEE International Conference on Microelectronic Test Structures*, 2009, p. 137-142.

About Authors ...

Miloslav KUBAŘ (*1981) received his M.Sc. degree in Electronics from the Czech Technical University (CTU), Prague, in 2007. He has worked in the company ASICentrum spol. s r.o. since 2007 as an analog IC designer. He also works toward Ph.D. in microsensor signal processing at the CTU. The topic of his thesis is novel optimization tool for analog integrated circuits design.

Jiří JAKOVENKO (*1972) graduated in Microelectronics from the Czech Technical University in Prague, Department of Microelectronics (FEE-CTU). Since 1996 he has been with FEE CTU, where he received PhD degree in 2004 in Electronics. Currently he works as an Assistant Professor at the Department of Microelectronics as a member of Microsystems group. His activities are oriented to MEMS design and modeling, design and development of RF ICs and SSL lightning. From 2005 to 2009 he was a consultant and a member of Cadence RF and Analog-Mix signal group, San Jose, USA. He is author of many scientific and technical papers, a member of the Programme and Organizing Committees of international scientific conferences. He is the leader of the IC design group at the Department of Microelectronics of the CTU.